IN THE SPECIFICATION:

Please amend the carryover paragraph between pages 9-10 of the Specification as shown below:

--FIGURE 5 is a block diagram representation illustrating in more detail the Image Acquisition Toolbox 230 in the illustrative embodiment of the present invention depicted in FIGURE 4. The Image Acquisition Toolbox 230 is divided into two layers including an adaptor layer 550 and an engine layer 560. The adaptor layer 550 includes an adaptor 430, a frame kit 530 and a data object (ImaqFrame) 540. The adaptor 430 receives data from the data source 110 through is driver 420. The adaptor layer 550 provides a frame kit 530 that includes a library of functions for creating and managing the data object 540. The functions in the frame kit 530 are called by the adaptor 430 to create the data object 540 into which the data received from the data source 110 is encapsulated. Additional metadata, such as time stamp, memory information and index in a sequence, may be stored in the data object 540, which can be made accessible to users and data sink objects 150-190 for display, processing, memory logging, file logging, etc. When the data received from the data source 110 is encapsulated into the data object 540, memory for the data object 540 is automatically allocated by the frame kit 530. The memory is also de-allocated or reused automatically by the frame kit 530 when the data object 540 is no longer used by data sink objects. Moreover, a maximum amount of memory may be configured for use by all data objects at a given instance in time.--

Please amend the paragraph beginning at page 10, line 22 of the Specification as shown below:

--Once the adaptor 430 calls the functions in the frame kit 530 to create the data object 540, the functions return identification information of the data object 540, such as a pointer of the data object 540. The pointer of the data object 540 indicates the location of the data object 540 in the memory of the computer systems. By using the pointer of the data object 540, the memory for the data object 540 can be dynamically allocated and deallocated. FIGURE 7 is the flow chart summarizing an exemplary operation of the adaptor layer 550 in the illustrative embodiment of the present invention. First, data is received from the data source 110 (step 710) and encapsulated into a data object 540 (step 730). The data object 540 is posted to the data server object 570 with the information identifying the data object 540 (step 750 ~~730~~).

Please amend the carryover paragraph between pages 12-13 of the Specification as shown below:

--FIGURE 9a is the flow chart showing the operation of the illustrative embodiment of the present invention in connection with the data sink counter of the data object 540 described in FIGURE 9. When a data object is created by a data object creator (FrameKit), the data object is posted to a data server object (step 980). If no data sink object is registered with the data server object (step 981), the data object is deleted (step 982). If data sink objects are registered with the data server object (step 981), the data object is instructed to increment its data sink counter (step 983). Information identifying the data object, such as the pointer of the data object in a memory, is provided to each of the data sink objects (step 984). Each of the data sink objects accesses the data object using the identification information of the data object (step 985). When a data sink object no longer needs to access the data object, the data

object is instructed to decrement the data sink counter ~~object~~ (step 986). If the counter is decreased to zero (step 987), the data object is removed from the memory of the computer system or alternatively, the data object is stored in a separate location of the memory (step 988).